Weekly report (2013.9.9 ~9.15)

Done

1) Continue learning Equalizer based on the eqPly example, which is shipped with the Equalizer distribution and serves as a reference implementation of an Equalizer-based application of medium complexity.

   As shown in Figure 1, eqPly classes are inherited from two namespaces: eq namespace and co namespace. Classes inherited from eq namespace are responsible for the rendering, while those inherited from co namespace are responsible for distributing data. Node, Pipe, Window and Channel corresponds to machine, separate thread, OpenGL rendering context and a set of task methods that contributes to frames respectively.

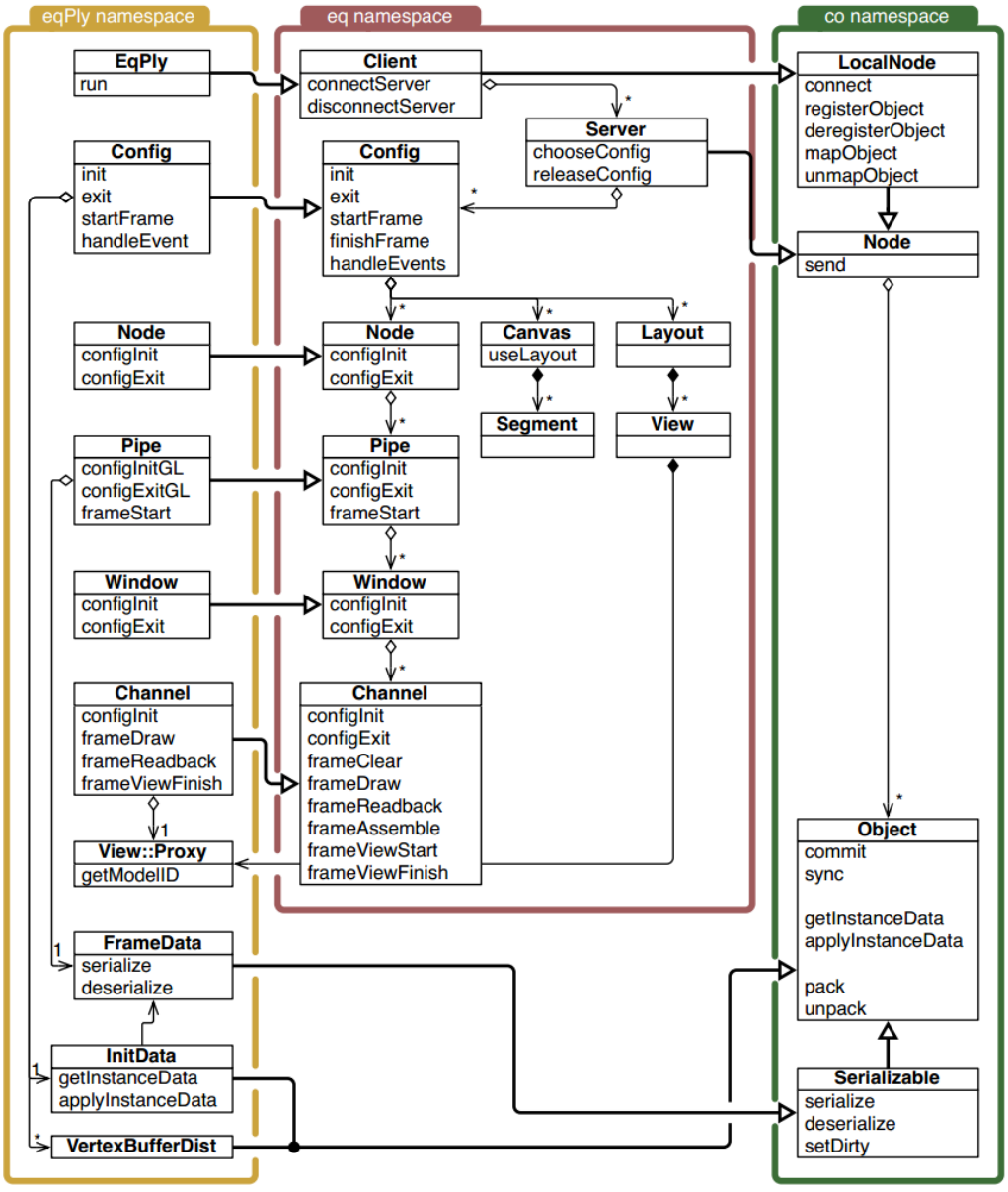

Figure 1 UML Diagram eqPly and relevant Equalizer Classes

I have understood the main work flow of eqPly, but not the detail of some special point (such as how each frame of an animation is controlled?). So, I plan to make those details

clear by implementing some basis of VisNgin (first, we plan to render an earth with texture parallelly based on Equalizer).

Haonan has already illustrated to me how earth rendering is implemented in VisNgin. I'll look into corresponding codes in VisNgin first.

2) I learned *cmake* made eqPlay a standalone vs project with it, adding essential dependency. So, we can focus on coding in the following development.
3) There comes the response for HPC and CAD Graphics, both accepted and supposed to be published, indexing by EI.

   I spent some time modifying the paper for HPC according to the reviewers' advises.

To Do

1) Reviews from both conference ask me to give more details of the MapReduce implementation and it's demand for the paper to be within 8 pages, so I need to adjust the content of both papers the next week.
2) Reading the code about rendering a textured earth in VisNgin.